Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

PCA9541

2-to-1 I^{2}C-bus master selector with interrupt logic and reset

Rev. 7.1 — 24 June 2015 **Product data sheet**

1. General description

The PCA9541 is a 2-to-1 I^{2}C-bus master selector designed for high reliability dual master I^{2}C-bus applications where system operation is required, even when one master fails or the controller card is removed for maintenance. The two masters (for example, primary and back-up) are located on separate I^{2}C-buses that connect to the same downstream I^{2}C-bus slave devices. I^{2}C-bus commands are sent by either I^{2}C-bus master and are used to select one master at a time. Either master at any time can gain control of the slave devices if the other master is disabled or removed from the system. The failed master is isolated from the system and will not affect communication between the on-line master and the slave devices on the downstream I^{2}C-bus.

Two versions are offered for different architectures. PCA9541/01 with channel 0 selected at start-up and PCA9541/03 with no channel selected after start-up.

The interrupt outputs are used to provide an indication of which master has control of the bus. One interrupt input ($\overline{\mathrm{INT\_IN}}$) collects downstream information and propagates it to the 2 upstream I^{2}C-buses ($\overline{\mathrm{INT0}}$ and $\overline{\mathrm{INT1}}$) if enabled. $\overline{\mathrm{INT0}}$ and $\overline{\mathrm{INT1}}$ are also used to let the previous bus master know that it is not in control of the bus anymore and to indicate the completion of the bus recovery/initialization sequence. Those interrupts can be disabled and will not generate an interrupt if the masking option is set.

A bus recovery/initialization if enabled sends nine clock pulses, a not acknowledge, and a STOP condition in order to set the downstream I^{2}C-bus devices to an initialized state before actually switching the channel to the selected master.

An interrupt is sent to the upstream channel when the recovery/initialization procedure is completed.

An internal bus sensor senses the downstream I^{2}C-bus traffic and generates an interrupt if a channel switch occurs during a non-idle bus condition. This function is enabled when the PCA9541 recovery/initialization is not used. The interrupt signal informs the master that an external I^{2}C-bus recovery/initialization needs to be performed. It can be disabled and an interrupt will not be generated.

The pass gates of the switches are constructed such that the V_{DD} pin can be used to limit the maximum high voltage, which will be passed by the PCA9541. This allows the use of different bus voltages on each pair, so that 1.8 V, 2.5 V, or 3.3 V devices can communicate with 5 V devices without any additional protection.

The PCA9541 does not isolate the capacitive loading on either side of the device, so the designer must take into account all trace and device capacitances on both sides of the device, and pull-up resistors must be used on all channels.

External pull-up resistors pull the bus to the desired voltage level for each channel. All I/O pins are 6.0 V tolerant.

An active LOW reset input allows the PCA9541 to be initialized. Pulling the $\overline{\text{RESET}}$ pin LOW resets the I^{2}C-bus state machine and configures the device to its default state as does the internal Power-On Reset (POR) function.

2. Features and benefits

- 2-to-1 bidirectional master selector
- I^{2}C-bus interface logic; compatible with SMBus standards
- PCA9541/01 powers up with Channel 0 selected
- PCA9541/03 powers up with no channel selected and either master can take control of the bus
- Active LOW interrupt input
- 2 active LOW interrupt outputs
- Active LOW reset input
- 4 address pins allowing up to 16 devices on the I^{2}C-bus
- Channel selection via I^{2}C-bus
- Bus initialization/recovery function
- Bus traffic sensor
- Low R_{on} switches
- Allows voltage level translation between 1.8 V, 2.5 V, 3.3 V and 5 V buses
- No glitch on power-up
- Supports hot insertion
- Software identical for both masters
- Low standby current
- Operating power supply voltage range of 2.3 V to 5.5 V
- 6.0 V tolerant inputs
- 0 Hz to 400 kHz clock frequency
- ESD protection exceeds 2000 V HBM per JESD22-A114, 200 V MM per JESD22-A115, and 1000 V CDM per JESD22-C101
- Latch-up testing is done to JEDEC Standard JESD78 which exceeds 100 mA
- Packages offered: SO16, TSSOP16, HVQFN16

3. Applications

- High reliability systems with dual masters
- Gatekeeper multiplexer on long single bus
- Bus initialization/recovery for slave devices without hardware reset
- Allows masters without arbitration logic to share resources

4. Ordering information

Table 1. Ordering information

T_{amb} = –40 °C to +85 °C

Type number	Package		
	Name	Description	Version
PCA9541D/01	SO16	plastic small outline package; 16 leads; body width 3.9 mm	SOT109-1
PCA9541PW/01	TSSOP16	plastic thin shrink small outline package; 16 leads; body width 4.4 mm	SOT403-1
PCA9541BS/01	HVQFN16	plastic thermal enhanced very thin quad flat package; no leads; 16 terminals; body 4 × 4 × 0.85 mm	SOT629-1
PCA9541D/03	SO16	plastic small outline package; 16 leads; body width 3.9 mm	SOT109-1
PCA9541PW/03	TSSOP16	plastic thin shrink small outline package; 16 leads; body width 4.4 mm	SOT403-1
PCA9541BS/03	HVQFN16	plastic thermal enhanced very thin quad flat package; no leads; 16 terminals; body 4 × 4 × 0.85 mm	SOT629-1

5. Marking

Table 2. Marking codes

Type number	Topside mark
PCA9541D/01	PCA9541D/01
PCA9541PW/01	9541/01
PCA9541BS/01	41/1
PCA9541D/03	PCA9541D/03
PCA9541PW/03	9541/03
PCA9541BS/03	41/3

6. Block diagram

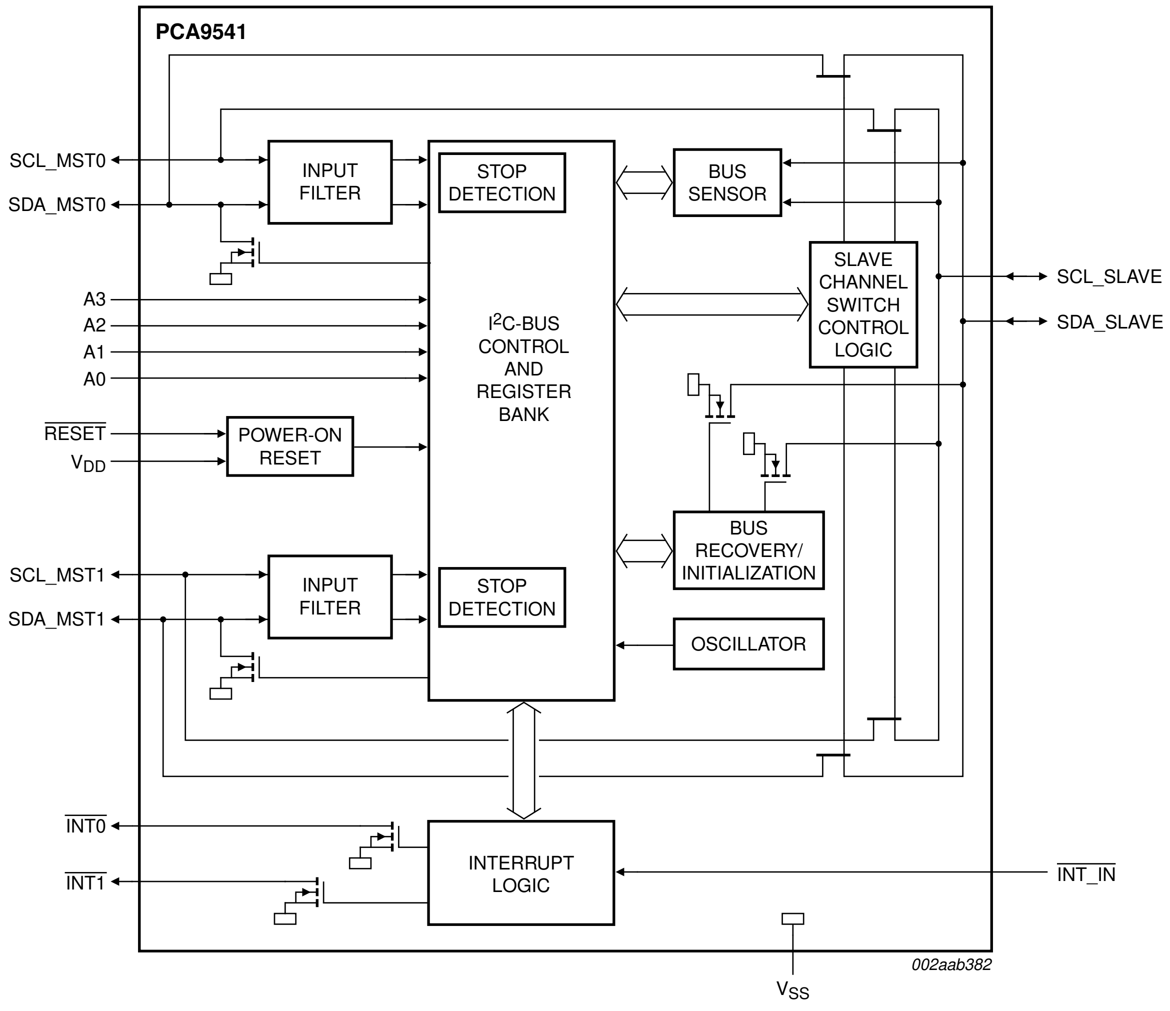


Fig 1. Block diagram of PCA9541

7. Pinning information

7.1 Pinning

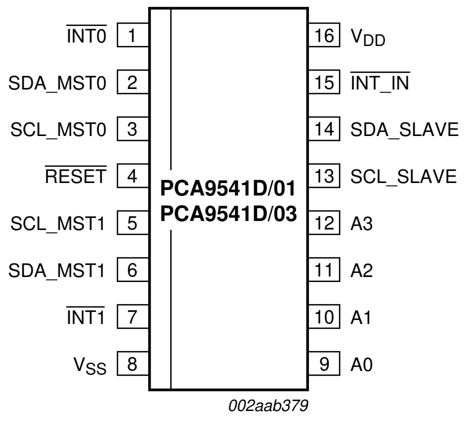


Fig 2. Pin configuration for SO16

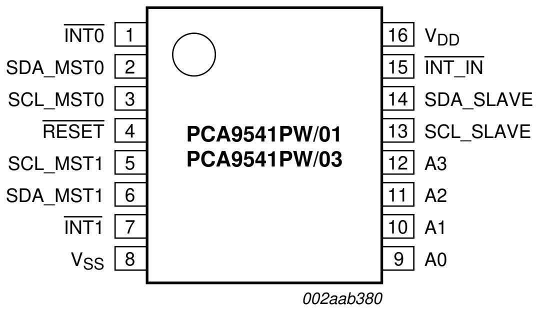


Fig 3. Pin configuration for TSSOP16

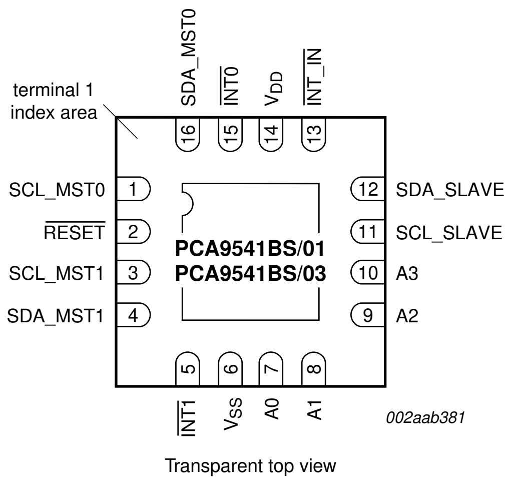


Fig 4. Pin configuration for HVQFN16

7.2 Pin description

Table 3. Pin description

Symbol	Pin		Description
	SO16, TSSOP16	HVQFN16	
$\overline{\text{INT0}}$	1	15	active LOW interrupt output 0 (external pull-up required)
SDA_MST0	2	16	serial data master 0 (external pull-up required)
SCL_MST0	3	1	serial clock master 0 (external pull-up required)
$\overline{\text{RESET}}$	4	2	active LOW reset input (external pull-up required)
SCL_MST1	5	3	serial clock master 1 (external pull-up required)
SDA_MST1	6	4	serial data master 1 (external pull-up required)
$\overline{\text{INT1}}$	7	5	active LOW interrupt output 1 (external pull-up required)
V_{SS}	8	6[1]	supply ground
A0	9	7	address input 0 (externally held to V_{SS} or V_{DD})
A1	10	8	address input 1 (externally held to V_{SS} or V_{DD})
A2	11	9	address input 2 (externally held to V_{SS} or V_{DD})
A3	12	10	address input 3 (externally held to V_{SS} or V_{DD})
SCL_SLAVE	13	11	serial clock slave (external pull-up required)
SDA_SLAVE	14	12	serial data slave (external pull-up required)
$\overline{\text{INT\_IN}}$	15	13	active LOW interrupt input (external pull-up required)
V_{DD}	16	14	supply voltage

[1] HVQFN16 package die supply ground is connected to both the V_{SS} pin and the exposed center pad. The V_{SS} pin must be connected to supply ground for proper device operation. For enhanced thermal, electrical, and board-level performance, the exposed pad needs to be soldered to the board using a corresponding thermal pad on the board, and for proper heat conduction through the board thermal vias need to be incorporated in the printed-circuit board in the thermal pad region.

8. Functional description

Refer to Figure 1 “Block diagram of PCA9541”.

8.1 Device address

Following a START condition, the upstream master that wants to control the I^{2}C-bus or make a status check must send the address of the slave it is accessing. The slave address of the PCA9541 is shown in Figure 5. To conserve power, no internal pull-up resistors are incorporated on the hardware selectable pins and they must be pulled HIGH or LOW.

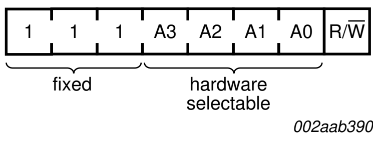


Fig 5. Slave address

The last bit of the slave address defines the operation to be performed. When set to logic 1 a read is selected, while logic 0 selects a write operation.

Remark: Reserved I^{2}C-bus addresses must be used with caution since they can interfere with:

- ‘reserved for future use’ I^{2}C-bus addresses (1111 1XX)
- slave devices that use the 10-bit addressing scheme (1111 0XX)

8.2 Command Code

Following the successful acknowledgement of the slave address, the bus master will send a byte to the PCA9541, which will be stored in the Command Code register.

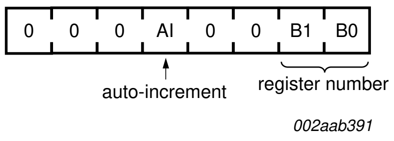


Fig 6. Command Code

The 2 LSBs are used as a pointer to determine which register will be accessed.

If the auto-increment flag is set (AI = 1), the two least significant bits of the Command Code are automatically incremented after a byte has been read or written. This allows the user to program the registers sequentially or to read them sequentially.

- During a read operation, the contents of these bits will roll over to 00b after the last allowed register is accessed (10b).

- During a write operation, the PCA9541 will acknowledge bytes sent to the IE and CONTROL registers, but will not acknowledge a byte sent to the Interrupt Status Register since it is a read-only register. The 2 LSBs of the Command Code do not roll over to 00b but stay at 10b.

Only the 2 least significant bits are affected by the AI flag.

Unused bits must be programmed with zeros. Any command code (write operation) different from ‘000AI 0000’, ‘000AI 0001’, and ‘000AI 0010’ will not be acknowledged. At power-up, this register defaults to all zeros.

Table 4. Command Code register

B1	B0	Register name	Type	Register function
0	0	IE	R/W	interrupt enable
0	1	CONTROL	R/W	control switch
1	0	ISTAT	R only	interrupt status
1	1	not allowed		

Each system master controls its own set of registers, however they can also read specific bits from the other system master.

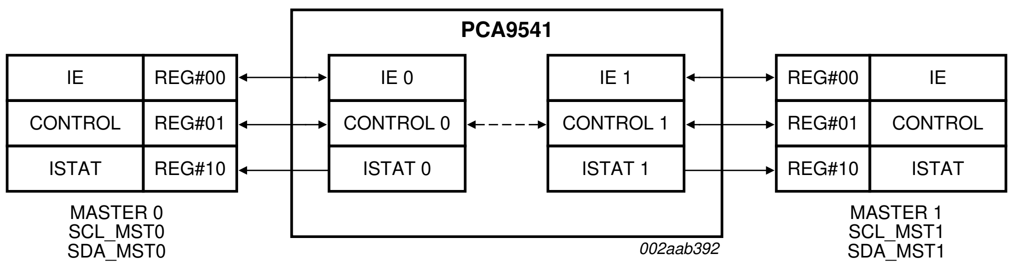


Fig 7. Internal register map

8.3 Interrupt Enable and Control registers description

When a master seeks control of the bus by connecting its I^{2}C-bus channel to the PCA9541 downstream channel, it has to write to the CONTROL register (Reg#01).

Bits MYBUS and BUSON allow the master to take control of the bus.

The MYBUS and the NMYBUS bits determine which master has control of the bus. Table 9 explains which master gets control of the bus and how. There is no arbitration. Any master can take control of the bus when it wants regardless of whether the other master is using it or not.

The BUSON and the NBUSON bits determine whether the upstream bus is connected or disconnected to/from the downstream bus. Table 10 explains when the upstream bus is connected or disconnected.

Internally, the state machine does the following:

- If the combination of the BUSON and the NBUSON bits causes the upstream to be disconnected from the downstream bus, then that is done. So in this case, the values of the MYBUS and the NMYBUS do not matter.

- If a master was connected to the downstream bus prior to the disconnect, then an interrupt is sent on the respective interrupt output in an attempt to let that master know that it is no longer connected to the downstream bus. This is indicated by setting the BUSLOST bit in the Interrupt Status Register.
- If the combination of the BUSON and the NBUSON bits causes a master to be connected to the downstream bus and if there is no change in the BUSON bits since when the disconnect took effect, then the master requesting the bus is connected to the downstream bus. If it requests a bus initialization sequence, then it is performed.
- If there is no change in the combination of the BUSON and the NBUSON bits and a new master wants the bus, then the downstream bus is disconnected from the old master that was using it and the new master gets control of it. Again, the bus initialization if requested is done. The appropriate interrupt signals are generated.

After a master has sent the bus control request:

1. The previous master is disconnected from the I^{2}C-bus. An interrupt to the previous master is sent through its $\overline{\text{INT}}$ line to let it know that it lost control of the bus. BUSLOST bit in the Interrupt Status Register is set. This interrupt can be masked by setting the BUSLOSTMSK bit to logic 1.
2. A built-in bus initialization/recovery function can take temporary control of the downstream channel to initialize the bus before making the actual switch to the new bus master. This function is activated by setting the BUSINIT to logic 1 by the master during the same write sequence as the one programming MYBUS and BUSON bits.

   When activated and whether the bus was previously idle or not:

   a. 9 clock pulses are sent on the SCL_SLAVE.

   b. SDA_SLAVE line is released (HIGH) when the clock pulses are sent to SCL_SLAVE. This is equivalent to sending 8 data bits and a not acknowledge.

   c. Finally a STOP condition is sent to the downstream slave channel.

   This sequence will complete any read transaction which was previously in process and the downstream slave configured as a slave-transmitter should release the SDA line because the PCA9541 did not acknowledge the last byte.
3. When the initialization has been requested and completed, the PCA9541 sends an interrupt to the new master through its $\overline{\text{INT}}$ line and connects the new master to the downstream channel. BUSINIT bit in the Interrupt Status Register is set. **The switch operation occurs after the master asking the bus control has sent a STOP command.** This interrupt can be masked by setting the BUSINITMSK bit to logic 1.
4. When the bus initialization/recovery function has not been requested (BUSINIT = 0), the PCA9541 connects the new master to the slave downstream channel. **The switch operation occurs after the master asking the bus control has sent a STOP command.** PCA9541 sends an interrupt to the new master through its $\overline{\text{INT}}$ line if the built-in bus sensor function detects a non-idle condition in the downstream slave channel at the switching time. BUSOK bit in the Interrupt Status Register is set. This means that a STOP condition has not been detected in the previous bus communication and that an external bus recovery/initialization must be performed. If an idle condition has been detected at the switching time, no interrupt will be sent. This interrupt can be masked by setting the BUSOKMSK bit to logic 1.

Interrupt status can be read. See Section 8.4 “Interrupt Status registers” for more information.

The MYTEST and the NMYTEST bits cause the interrupt pins of the respective masters to be activated for a 'functional interrupt test'.

Remark: The regular way to proceed is that a master asks to take the control of the bus by programming MYBUS and BUSON bits based on NMUYBUS and NBUSON values. Nevertheless, the same master can also decide to give up the control of the bus and give it to the other master. This is also done by programming the MYBUS and BUSON bits based on NMYBUS and NBUSON values.

Remark: Any writes either to the Interrupt Enable Register or the Control Register cause the respective register to be updated on the 9th clock cycle, that is, on the rising edge of the acknowledge clock cycle.

Remark: The actual switch from one channel to another or the switching off of both the channels happens on a STOP command that is sent by the master requesting the switch.

8.3.1 Register 0: Interrupt Enable (IE) register (B1:B0 = 00b)

This register allows a master to read and/or write (if needed) Mask options for its own channel.

The Interrupt Enable register described below is identical for both the masters. Nevertheless, there are physically 2 internal Interrupt Enable registers, one for each upstream channel. When Master 0 reads/writes in this register, the internal Interrupt Enable Register 0 will be accessed. When Master 1 reads/writes in this register, the internal Interrupt Enable Register 1 will be accessed.

Table 5. Register 0 - Interrupt Enable (IE) register (B1:B0 = 00b) bit allocation

7	6	5	4	3	2	1	0
0	0	0	0	BUSLOSTMSK	BUSOKMSK	BUSINITMSK	INTINMSK

Table 6. Register 0 - Interrupt Enable (IE) register bit description
*Legend: * default value*

Bit	Symbol	Access	Value[1]	Description
7:4	-	R only	0*	not used
3	BUSLOSTMSK	R/W	0*	An interrupt on $\overline{\text{INT}}$ will be generated after the other master has been disconnected.
			1	An interrupt on $\overline{\text{INT}}$ will not be generated after the other master has been disconnected.
2	BUSOKMSK	R/W	0*	After connection is requested and Bus Initialization not requested (BUSINIT = 0), an interrupt on $\overline{\text{INT}}$ will be generated when a non-idle situation has been detected on the downstream slave channel by the bus sensor at the switching moment. **Remark:** Channel switching is done automatically after the STOP command.
			1	After connection is requested and Bus Initialization not requested (BUSINIT = 0), an interrupt on $\overline{\text{INT}}$ will not be generated when a non-idle situation has been detected on the downstream slave channel by the bus sensor at the switching moment (masked). **Remark:** Channel switching is done automatically after the STOP command.

Table 6. Register 0 - Interrupt Enable (IE) register bit description *…continued*
*Legend: * default value*

Bit	Symbol	Access	Value[1]	Description
1	BUSINITMSK	R/W	0*	After connection is requested and Bus Initialization requested (BUSINIT = 1), an interrupt on $\overline{\text{INT}}$ will be generated when the bus initialization is done. **Remark:** Channel switching is done after bus initialization completed.
			1	After connection is requested and Bus Initialization requested (BUSINIT = 1), an interrupt on $\overline{\text{INT}}$ will not be generated when the bus initialization is done (masked). **Remark:** Channel switching is done after bus initialization completed.
0	INTINMSK	R/W	0*	Interrupt on $\overline{\text{INT\_IN}}$ will generate an interrupt on $\overline{\text{INT}}$.
			1	Interrupt on $\overline{\text{INT\_IN}}$ will not generate an interrupt on $\overline{\text{INT}}$ (masked)

[1] Default values are the same for PCA9541/01 and PCA9541/03.

8.3.2 Register 1: Control Register (B1:B0 = 01b)

The Control Register described below is identical for both the masters. Nevertheless, there are physically 2 internal Control Registers, one for each upstream channel. When master 0 reads/writes in this register, the internal Control Register 0 will be accessed. When master 1 reads/writes in this register, the internal Control Register 1 will be accessed.

Table 7. Register 1 - Control Register (B1:B0 = 01b) bit allocation

7	6	5	4	3	2	1	0
NTESTON	TESTON	0	BUSINIT	NBUSON	BUSON	NMYBUS	MYBUS

Table 8. Register 1 - Control Register (B1:B0 = 01b) bit description
*Legend: * default value*

Bit	Symbol	Access	Value[1]	Description
7	NTESTON	R/W	0*	A logic level HIGH to the $\overline{\text{INT}}$ line of the other channel is sent (interrupt cleared).
			1	A logic level LOW to the $\overline{\text{INT}}$ line of the other channel is sent (interrupt generated).
6	TESTON	R/W	0*	A logic level HIGH to the $\overline{\text{INT}}$ line is sent (interrupt cleared).
			1	A logic level LOW to the $\overline{\text{INT}}$ line is sent (interrupt generated).
5	-	R only	0*	not used
4	BUSINIT	R/W	0*	Bus initialization is not requested.
			1	Bus initialization is requested.
3	NBUSON	R only	see Table 11	NBUSON bit along with BUSON bit decides whether any upstream channel is connected to the downstream channel or not. See Table 10, Table 11, and Table 12.
2	BUSON	R/W	see Table 11	BUSON bit along with the NBUSON bit decides whether any upstream channel is connected to the downstream channel or not. See Table 10, Table 11, and Table 12.
1	NMYBUS	R only	see Table 11	NMYBUS bit along with MYBUS bit decides which upstream channel is connected to the downstream channel. See Table 9, Table 11, and Table 12.
0	MYBUS	R/W	see Table 11	MYBUS bit along with the NMYBUS bit decides which upstream channel is connected to the downstream channel. See Table 9, Table 11, and Table 12.

[1] Default values are the same for PCA9541/01 and PCA9541/03.

Table 9. MYBUS and NMYBUS truth table
As a master reads its Control Register

NMYBUS[1]	MYBUS[1]	Slave channel
0	0	The master reading this combination has control of the bus.
1	0	The master reading this combination does not have control of the bus.
0	1	The master reading this combination does not have control of the bus.
1	1	The master reading this combination has control of the bus.

[1] MYBUS and NMYBUS is an exclusive-OR type function where:
Equal values (00b or 11b) means that the master reading its Control Register has control of the bus.
Different values (01b or 10b) means that the master reading its Control Register does not have control of the bus.

Table 10. BUSON and NBUSON truth table

NBUSON[1]	BUSON[1]	Slave channel
0	0	off
1	0	on
0	1	on
1	1	off

[1] BUSON and NBUSON is an exclusive-OR type function where:
Equal values (00b or 11b) means that the connection between the upstream and the downstream channels is off.
Different values (01b or 10b) means that the connection between the upstream and the downstream channels is on.

Switch to the new channel is done when the master initiating the switch request sends a STOP command to the PCA9541.

If either master wants to change the connection of the downstream channel, it needs to write to **its Control Register (Reg#01), and then send a STOP command** because an update of the connection to the downstream according to the values in the two internal Control Registers happens only on a STOP command. Writing to one control register followed by a STOP condition on the other master's channel will not cause an update to the downstream connection.

When both masters request a switch to their own channel at the same time, the master who last wrote to its Control Register before the PCA9541 receives a STOP command wins the switching sequence. There is no arbitration performed.

The Auto Increment feature (AI = 1) allows to program the PCA9541 in 4 bytes:

```
Start
111A3A2A1A0 + 0     PCA9541 Address + Write
00010000            Select Reg#00 with AI = 1
Data Reg#00         Interrupt Enable Register data
Data Reg#01         Control Register data
Stop
```

Table 11. Default Control Register values

Type version	Master	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		NTESTON	TESTON	not used	BUSINIT	NBUSON	BUSON	NMYBUS	MYBUS
PCA9541/01	MST_0	0	0	0	0	0	1	0	0
	MST_1	0	0	0	0	1	0	1	0
PCA9541/03	MST_0	0	0	0	0	0	0	0	0
	MST_1	0	0	0	0	0	0	1	0

Table 12 describes which command needs to be written to the Control Register when a master device wants to take control of the I^{2}C-bus. Byte written to the Control Register is a function of the current I^{2}C-bus control status performed after an initial reading of the Control Register.

Current status of the I^{2}C-bus is determined by the bits MYBUS, NMYBUS, BUSON and NBUSON is one of the following:

- The master reading its Control Register does not have control and the I^{2}C-bus is off.
- The master reading its Control Register does not have control and the I^{2}C-bus is on.
- The master reading its Control Register has control and the I^{2}C-bus is off.
- The master reading its Control Register has control and the I^{2}C-bus is on.

'I^{2}C-bus off' means that upstream and downstream channels are not connected together.

'I^{2}C-bus on' means that upstream and downstream channels are connected together.

Remark: Only the 4 LSBs of the Control Register are described in Table 12 since only those bits control the I^{2}C-bus control. The logic value for the 4 MSBs is specific to the application and are not discussed in the table.

The read sequence is performed by the master as:
S - 111xxxx0 - 000x0001 - Sr - 111xxxx1 - DataRead - P

The write sequence is performed by the master as:
S - 111xxxx0 - 000x0001 - DataWritten - P

Table 12. Bus control sequence

Read Control Register performed by the master							Write Control Register performed by the master					
Byte read[1] Hex	Status		NBUSON	BUSON	NMYBUS	MYBUS	Byte written[1][2] Hex	Action performed to take mastership	NBUSON[3]	BUSON	NMYBUS[3]	MYBUS
0	bus off	has control	0	0	0	0	4	bus on	x	1	x	0
1	bus off	no control	0	0	0	1	4	bus on, take control	x	1	x	0
2	bus off	no control	0	0	1	0	5	bus on, take control	x	1	x	1
3	bus off	has control	0	0	1	1	5	bus on	x	1	x	1
4	bus on	has control	0	1	0	0	-	no change	no write required			
5	bus on	no control	0	1	0	1	4	take control	x	1	x	0
6	bus on	no control	0	1	1	0	5	take control	x	1	x	1
7	bus on	has control	0	1	1	1	-	no change	no write required			
8	bus on	has control	1	0	0	0	-	no change	no write required			
9	bus on	no control	1	0	0	1	0	take control	x	0	x	0
A	bus on	no control	1	0	1	0	1	take control	x	0	x	1
B	bus on	has control	1	0	1	1	-	no change	no write required			
C	bus off	has control	1	1	0	0	0	bus on	x	0	x	0
D	bus off	no control	1	1	0	1	0	bus on, take control	x	0	x	0
E	bus off	no control	1	1	1	0	1	bus on, take control	x	0	x	1
F	bus off	has control	1	1	1	1	1	bus on	x	0	x	1

[1] Only the 4 LSBs are shown.

[2] x0x0 in binary = 0, 2, 8 or A in hexadecimal
x0x1 in binary = 1, 3, 9 or B in hexadecimal
x1x0 in binary = 4, 6, C or E in hexadecimal
x1x1 in binary = 5, 7, D or F in hexadecimal

[3] x can be either ‘0’ or ‘1’ since those bits are read-only bits.

8.4 Interrupt Status registers

The PCA9541 provides 4 different types of interrupt:

- To indicate to the former I²C-bus master that it is not in control of the bus anymore
- To indicate to the new I²C-bus master that:
  - The bus recovery/initialization has been performed and that the downstream channel connection has been done (built-in bus recovery/initialization active).
  - A 'bus not well initialized' condition has been detected by the PCA9541 when the switch has been done (built-in bus recovery/initialization not active). This information can be used by the new master to initiate its own bus recovery/initialization sequence.
- Indicate to both I²C-bus upstream masters that a downstream interrupt has been generated through the $\overline{\text{INT\_IN}}$ pin.
- Functionality wiring test.

8.4.1 Bus control lost interrupt

When an upstream master takes control of the I²C-bus while the other channel was using the downstream channel, an interrupt is generated to the master losing control of the bus ($\overline{\text{INT}}$ line goes LOW to let the master know that it lost the control of the bus) immediately after disconnection from the downstream channel.

By setting the BUSLOSTMSK bit to '1', the interrupt is masked and the upstream master that lost the I²C-bus control does not receive an interrupt ($\overline{\text{INT}}$ line does not go LOW).

8.4.2 Recovery/initialization interrupt

Before switching to a new upstream channel, an automatic bus recovery/initialization can be performed by the PCA9541. This function is requested by setting the BUSINIT bit to '1'. When the downstream bus has been initialized, an interrupt to the new master is generated ($\overline{\text{INT}}$ line goes LOW).

By setting the BUSINITMSK bit to '1', the interrupt is masked and the new master does not receive an interrupt ($\overline{\text{INT}}$ line does not go LOW).

When the automatic bus recovery/initialization is not requested, if the built-in bus sensor function (sensing permanently the downstream I²C-bus traffic) detects a non-idle condition (previous bus channel connected to the downstream slave channel, was between a START and STOP condition), then an interrupt to the new master is sent ($\overline{\text{INT}}$ line goes LOW). This interrupt tells the new master that an external bus recovery/initialization must be performed. By setting the BUSOKMSK bit to '1', the interrupt is masked and the new master does not receive an interrupt ($\overline{\text{INT}}$ line does not go LOW).

Remark: In this particular situation, after the switch to the new master is performed, **a read of the Interrupt Status Register is not possible if the switch happened in the middle of a read sequence** because the new master does not have control of the SDA line.

8.4.3 Downstream interrupt

An interrupt can also be generated by a downstream device by asserting the $\overline{\text{INT\_IN}}$ pin LOW. When $\overline{\text{INT\_IN}}$ is asserted LOW and if both INTINMSK bits are not set to '1' by either master, $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ both go LOW.

By setting the INTINMSK bit to '1' by a master and/or the INTINMSK bit to '1' by the other master, the interrupt(s) is (are) masked and the corresponding masked channel(s) does (do) not receive an interrupt ($\overline{\text{INT0}}$ and/or $\overline{\text{INT1}}$ line does (do) not go LOW).

8.4.4 Functional test interrupt

A master can send an interrupt to itself to test its own $\overline{\text{INT}}$ wire or send an interrupt to the other master to test its $\overline{\text{INT}}$ line. This is done by:

- setting the TESTON bit to '1' to test its own $\overline{\text{INT}}$ line
- setting the NTESTON bit to '1' to test the other master $\overline{\text{INT}}$ line

Setting the TESTON and/or NTESTON bits to '0' by a master will clear the interrupt(s).

Remark: Interrupt outputs have an open-drain structure. Interrupt input does not have any internal pull-up resistor and must not be left floating (that is, pulled HIGH to V_{DD} through resistor) in order to avoid any undesired interrupt conditions.

8.4.5 Register 2: Interrupt Status Register (B1:B0 = 10b)

The Interrupt Status Register for both the masters is identical and is described below. Nevertheless, there are physically 2 internal Interrupt Registers, one for each upstream channel.

When Master 0 reads this register, the internal Interrupt Register 0 will be accessed.

When Master 1 reads this register, the internal Interrupt Register 1 will be accessed.

Table 13. Register 2 - Interrupt Status register (B1:B0 = 10b) bit allocation

7	6	5	4	3	2	1	0
NMYTEST	MYTEST	0	0	BUSLOST	BUSOK	BUSINIT	INTIN

Table 14. Register 2 - Interrupt Status (ISTAT) register bit description
*Legend: * default value*

Bit	Symbol	Access	Value[1]	Description
7	NMYTEST[2]	R only	0*	no interrupt generated due to NTESTON bit from the other master (NTESTON = 0 from the other master)[3]
			1	interrupt generated due to TESTON bit from the other master (NTESTON = 1 from the other master)[3]
6	MYTEST[2]	R only	0*	no interrupt generated by TESTON bit (TESTON = 0)[3]
			1	interrupt generated by TESTON bit (TESTON = 1)[3]
5	-	R only	0*	not used
4	-	R only	0*	not used
3	BUSLOST[4]	R only	0*	no interrupt generated to the previous master when switching to the new one is initiated
			1	interrupt generated to the previous master when switching to the new one is initiated

Table 14. Register 2 - Interrupt Status (ISTAT) register bit description *…continued*
*Legend: * default value*

Bit	Symbol	Access	Value[1]	Description
2	BUSOK[4]	R only	0*	no interrupt generated by bus sensor function
			1	interrupt generated by bus sensor function (masked when bus recovery/initialization requested) - Bus was not idle when the switch occurred
1	BUSINIT[4]	R only	0*	no interrupt generated by the bus recovery/initialization function
			1	interrupt generated by the bus recovery/initialization function; recovery/initialization done
0	INTIN[2]	R only	0*	no interrupt on interrupt input ($\overline{\text{INT\_IN}}$)[5]
			1	interrupt on interrupt input ($\overline{\text{INT\_IN}}$)[5]

[1] Default values are the same for PCA9541/01 and PCA9541/03.

[2] Reading the Interrupt Status Register does not clear the MYTEST, NMYTEST or the INTIN bits. They are cleared if:
$\overline{\text{INT\_IN}}$ lines goes HIGH for INTIN bit
TESTON bit is cleared for MYTEST bit
NTESTON bit is cleared for NMYTEST bit

[3] Interrupt on a master is cleared after TESTON bit is cleared by the same master or NTESTON bit is cleared by the other master.

[4] BUSINIT, BUSOK and BUSLOST bits in the Interrupt Status Register get cleared after a read of the same register is done. Precisely, the register gets cleared on the second clock pulse during the read operation.

[5] If the interrupt condition remains on $\overline{\text{INT\_IN}}$ after the read sequence, another interrupt will be generated (if the interrupt has not been masked).

8.5 Power-on reset

When power is applied to V_{DD}, an internal power-on reset holds the PCA9541 in a reset condition until V_{DD} has reached V_{POR}. At this point, the reset condition is released and the internal registers are initialized to their default states, with:

- PCA9541/01: default Channel 0 (no STOP detect)

  After power-up and/or insertion of the device in the main I^{2}C-bus, the upstream Channel 0 and the downstream slave channel are connected together.

- PCA9541/03: default ‘no channel’ (no STOP detect)

  After power-up and/or insertion of the device in the main I^{2}C-bus, no channel will be connected to the downstream channel. The device is ready to receive a START condition and its address by a master.

  If either register writes to its Control Register, then the connection between the upstream and the downstream channels is determined by the values on the Control Registers.

Thereafter, V_{DD} must be lowered below 0.2 V to reset the device.

8.6 External reset

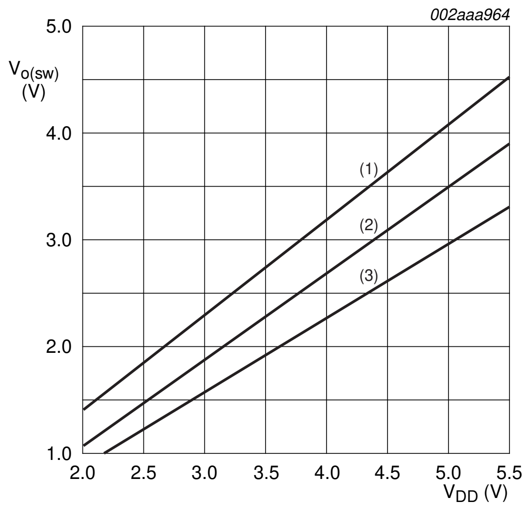
A reset can be accomplished by holding the $\overline{\text{RESET}}$ pin LOW for a minimum of $t_{w(rst)L}$. The PCA9541 registers and I²C-bus state machine will be held in their default states until the $\overline{\text{RESET}}$ input is once again HIGH. This input typically requires a pull-up resistor to V_{DD}.

Default states are:

- I²C-bus upstream Channel 0 connected to the I²C-bus downstream channel for the PCA9541/01
- no I²C-bus upstream channel connected to the I²C-bus downstream channel for the PCA9541/03.

8.7 Voltage translation

The pass gate transistors of the PCA9541 are constructed such that the V_{DD} voltage can be used to limit the maximum voltage that will be passed from one I²C-bus to another.



(1) maximum

(2) typical

(3) minimum

Fig 8. Pass gate voltage as a function of supply voltage

Figure 8 shows the voltage characteristics of the pass gate transistors (note that the graph was generated using the data specified in Section 12 “Static characteristics” of this data sheet). In order for the PCA9541 to act as a voltage translator, the $V_{o(sw)}$ voltage should be equal to, or lower than the lowest bus voltage. For example, if the main buses were running at 5 V, and the downstream bus was 3.3 V, then $V_{o(sw)}$ should be equal to or below 3.3 V to effectively clamp the downstream bus voltages. Looking at Figure 8, we see that $V_{o(sw)(max)}$ will be at 3.3 V when the PCA9541 supply voltage is 3.5 V or lower so the PCA9541 supply voltage could be set to 3.3 V. Pull-up resistors can then be used to bring the bus voltages to their appropriate levels (see Figure 17).

More Information on voltage translation can be found in Application Note *AN262: PCA954X family of I²C/SMBus multiplexers and switches*.

9. Characteristics of the I^{2}C-bus

The I^{2}C-bus is for 2-way, 2-line communication between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). Both lines must be connected to a positive supply via a pull-up resistor when connected to the output stages of a device. Data transfer may be initiated only when the bus is not busy.

9.1 Bit transfer

One data bit is transferred during each clock pulse. The data on the SDA line must remain stable during the HIGH period of the clock pulse as changes in the data line at this time will be interpreted as control signals (see Figure 9).

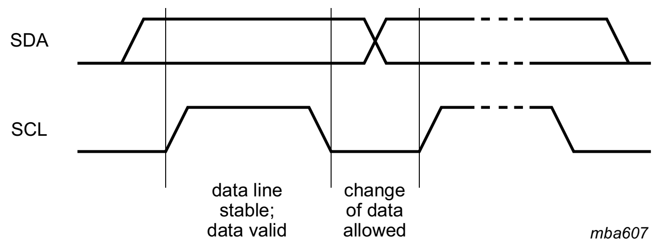


Fig 9. Bit transfer

9.2 START and STOP conditions

Both data and clock lines remain HIGH when the bus is not busy. A HIGH-to-LOW transition of the data line, while the clock is HIGH is defined as the START condition (S). A LOW-to-HIGH transition of the data line while the clock is HIGH is defined as the STOP condition (P) (see Figure 10).

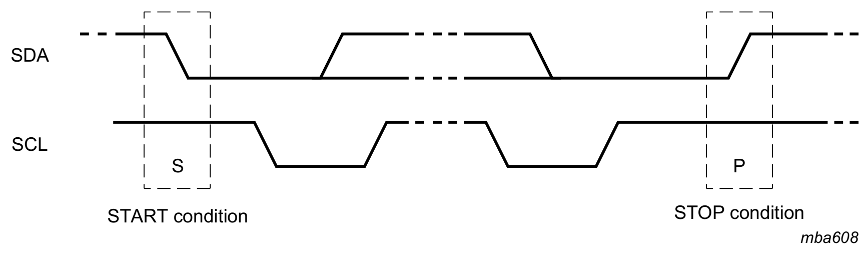


Fig 10. Definition of START and STOP conditions

9.3 System configuration

A device generating a message is a ‘transmitter’, a device receiving is the ‘receiver’. The device that controls the message is the ‘master’ and the devices which are controlled by the master are the ‘slaves’ (see Figure 11).

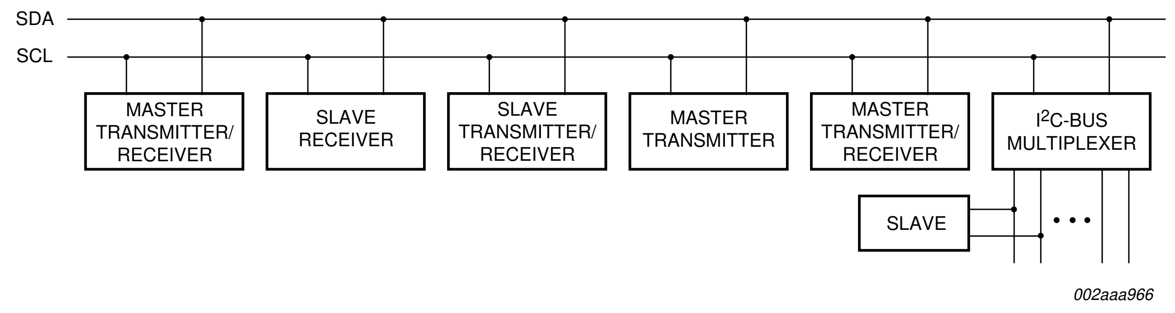

SDA
SCL
MASTER TRANSMITTER/ RECEIVER
SLAVE RECEIVER
SLAVE TRANSMITTER/ RECEIVER
MASTER TRANSMITTER
MASTER TRANSMITTER/ RECEIVER
I²C-BUS MULTIPLEXER
SLAVE
002aaa966

Fig 11. System configuration

9.4 Acknowledge

The number of data bytes transferred between the START and the STOP conditions from transmitter to receiver is not limited. Each byte of eight bits is followed by one acknowledge bit. The acknowledge bit is a HIGH level put on the bus by the transmitter, whereas the master generates an extra acknowledge related clock pulse.

A slave receiver which is addressed must generate an acknowledge after the reception of each byte. Also a master must generate an acknowledge after the reception of each byte that has been clocked out of the slave transmitter. The device that acknowledges has to pull down the SDA line during the acknowledge clock pulse, so that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse; set-up and hold times must be taken into account.

A master receiver must signal an end of data to the transmitter by not generating an acknowledge on the last byte that has been clocked out of the slave. In this event, the transmitter must leave the data line HIGH to enable the master to generate a STOP condition.

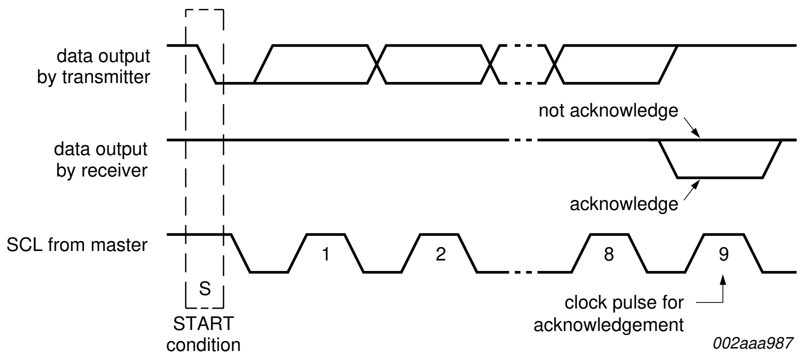


Fig 12. Acknowledgement on the I²C-bus

9.5 Bus transactions

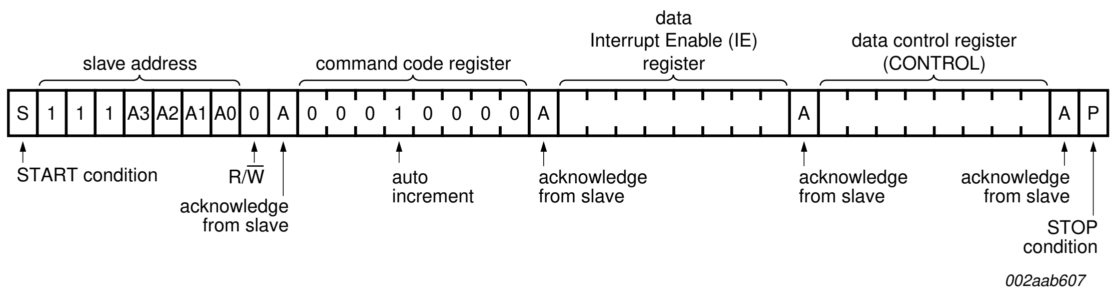
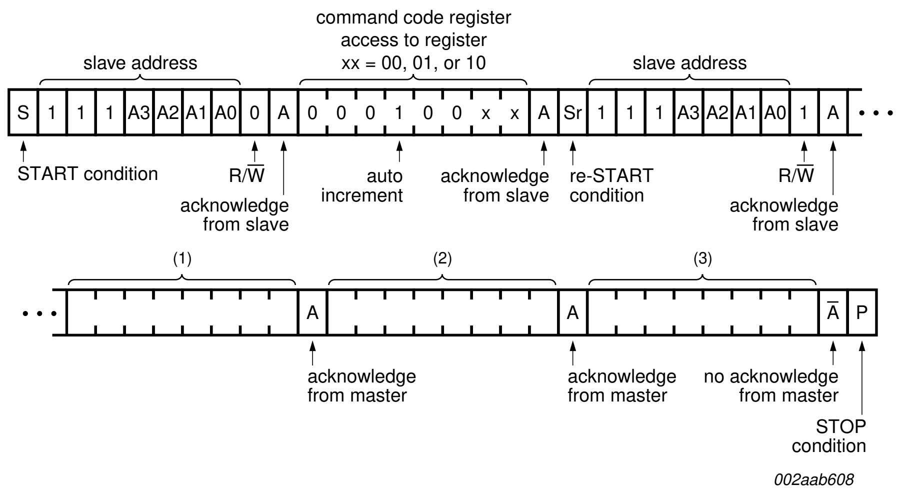


Fig 13. Write to the Interrupt Enable and Control registers using the Auto-Increment (AI) bit

Remark: If a third data byte is sent, it will not be acknowledged by the PCA9541.



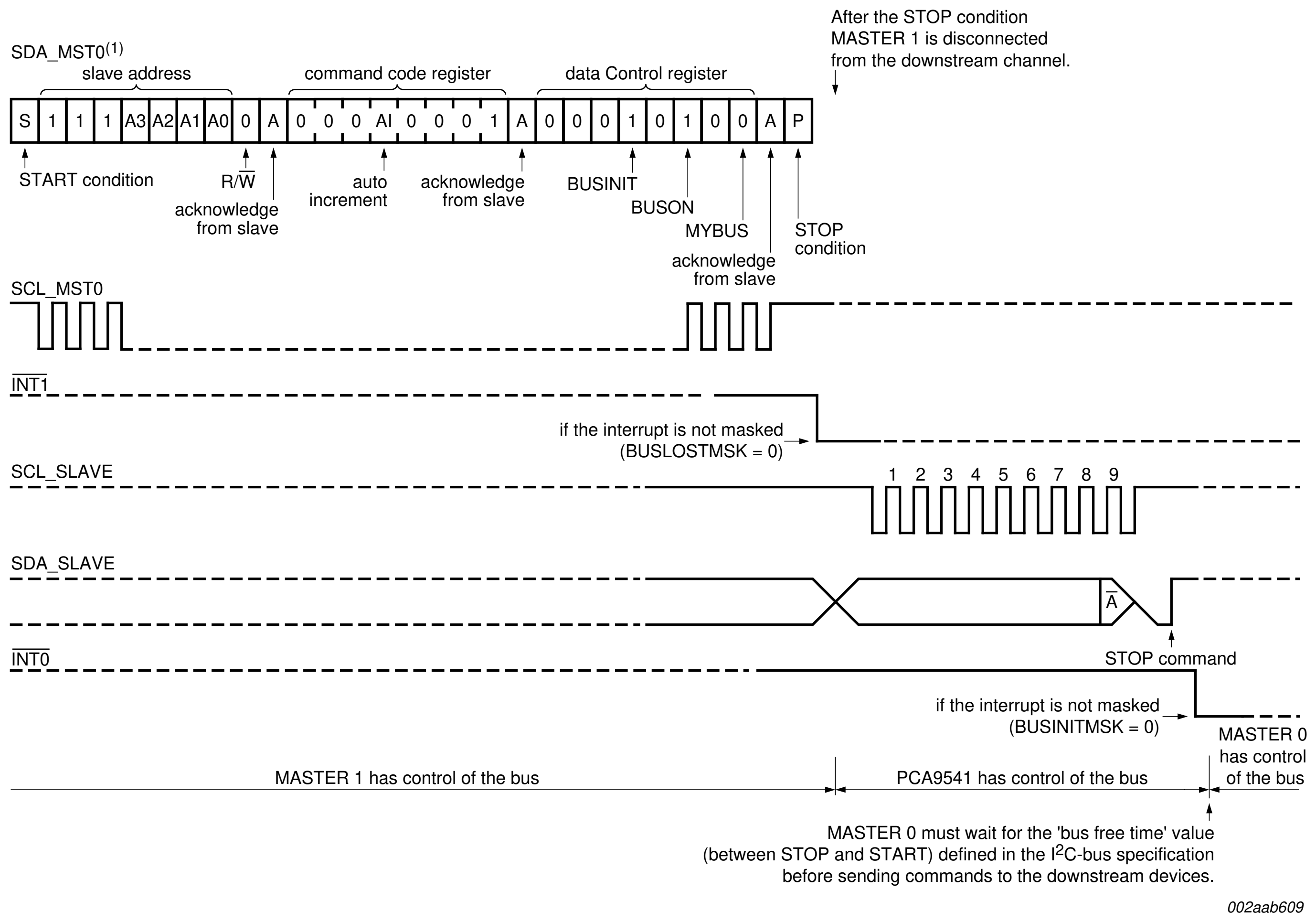
(1) xx = 00: Interrupt Enable register
xx = 01: Control register
xx = 10: INT register

(2) xx = 00: Control register
xx = 01: INT register
xx = 10: Interrupt Enable register

(3) xx = 00: INT register
xx = 01: Interrupt Enable register
xx = 10: Control register

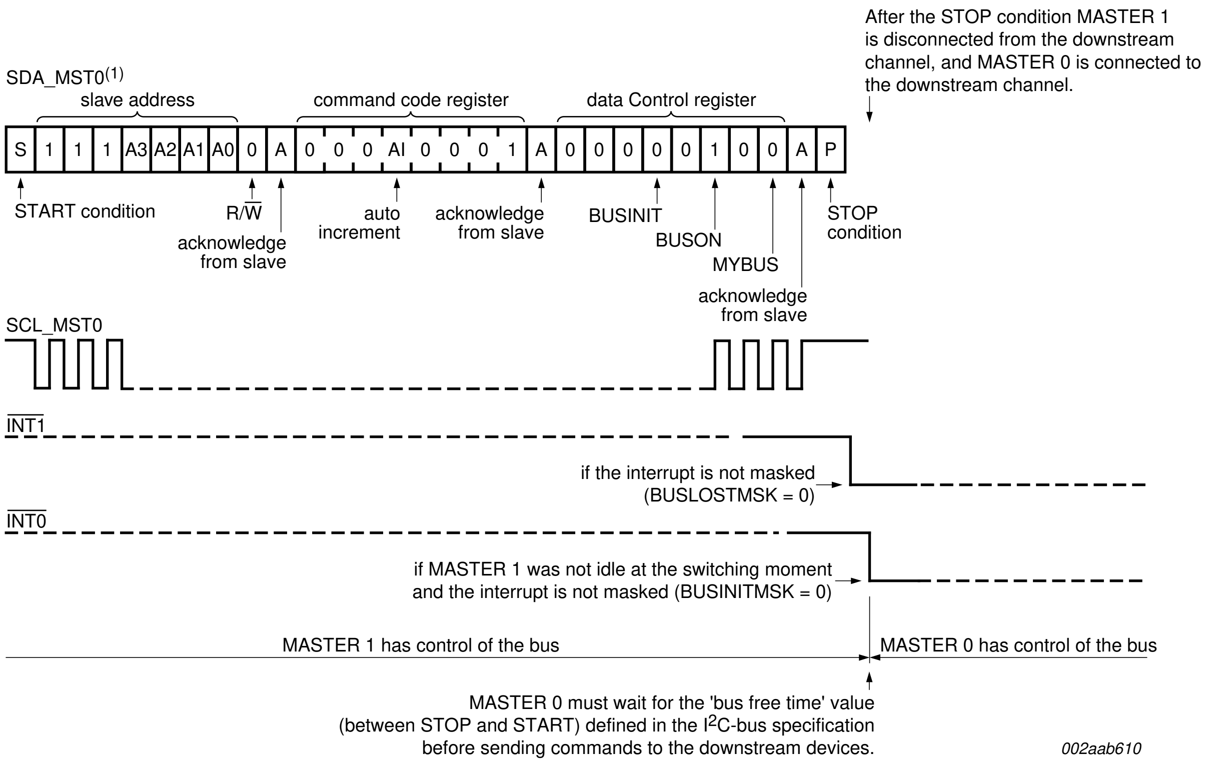
Fig 14. Read the 3 registers using the Auto-Increment (AI) bit

Remark: If a fourth data byte is read, the first register will be accessed.


After the STOP condition MASTER 1 is disconnected from the downstream channel.
SDA_MST0(1)
slave address
command code register
data Control register
S 1 1 1 A3 A2 A1 A0 0 A 0 0 0 AI 0 0 0 1 A 0 0 0 1 0 1 0 0 A P
START condition
R/W
acknowledge from slave
auto increment
acknowledge from slave
BUSINIT
BUSON
MYBUS
acknowledge from slave
STOP condition
SCL_MST0
INT1
if the interrupt is not masked (BUSLOSTMSK = 0)
SCL_SLAVE
1 2 3 4 5 6 7 8 9
SDA_SLAVE
A
STOP command
INT0
if the interrupt is not masked (BUSINITMSK = 0)
MASTER 0 has control of the bus
MASTER 1 has control of the bus
PCA9541 has control of the bus
MASTER 0 must wait for the 'bus free time' value (between STOP and START) defined in the I2C-bus specification before sending commands to the downstream devices.
002aab609

(1) We assume that a read of the Control register was done by MASTER 0 before this sequence and that 000x 0101 was read (MASTER 1 controlling the bus).

Fig 15. Write to the Control register and switch from Channel 1 to Channel 0 (bus recovery/initialization requested)


After the STOP condition MASTER 1 is disconnected from the downstream channel, and MASTER 0 is connected to the downstream channel.
SDA_MST0(1)
slave address
command code register
data Control register
S 1 1 1 A3 A2 A1 A0 0 A 0 0 0 AI 0 0 0 1 A 0 0 0 0 0 1 0 0 A P
START condition
R/W
acknowledge from slave
auto increment
acknowledge from slave
BUSINIT
BUSON
MYBUS
acknowledge from slave
STOP condition
SCL_MST0
INT1
if the interrupt is not masked (BUSLOSTMSK = 0)
INT0
if MASTER 1 was not idle at the switching moment and the interrupt is not masked (BUSINITMSK = 0)
MASTER 1 has control of the bus
MASTER 0 has control of the bus
MASTER 0 must wait for the 'bus free time' value (between STOP and START) defined in the I²C-bus specification before sending commands to the downstream devices.
002aab610

(1) We assume that a read of the Control register was done by MASTER 0 before this sequence and that 000x 0101 was read (MASTER 1 controlling the bus).

Fig 16. Write to the Control register and switch from Channel 1 to Channel 0 (bus recovery/initialization not requested)

10. Application design-in information

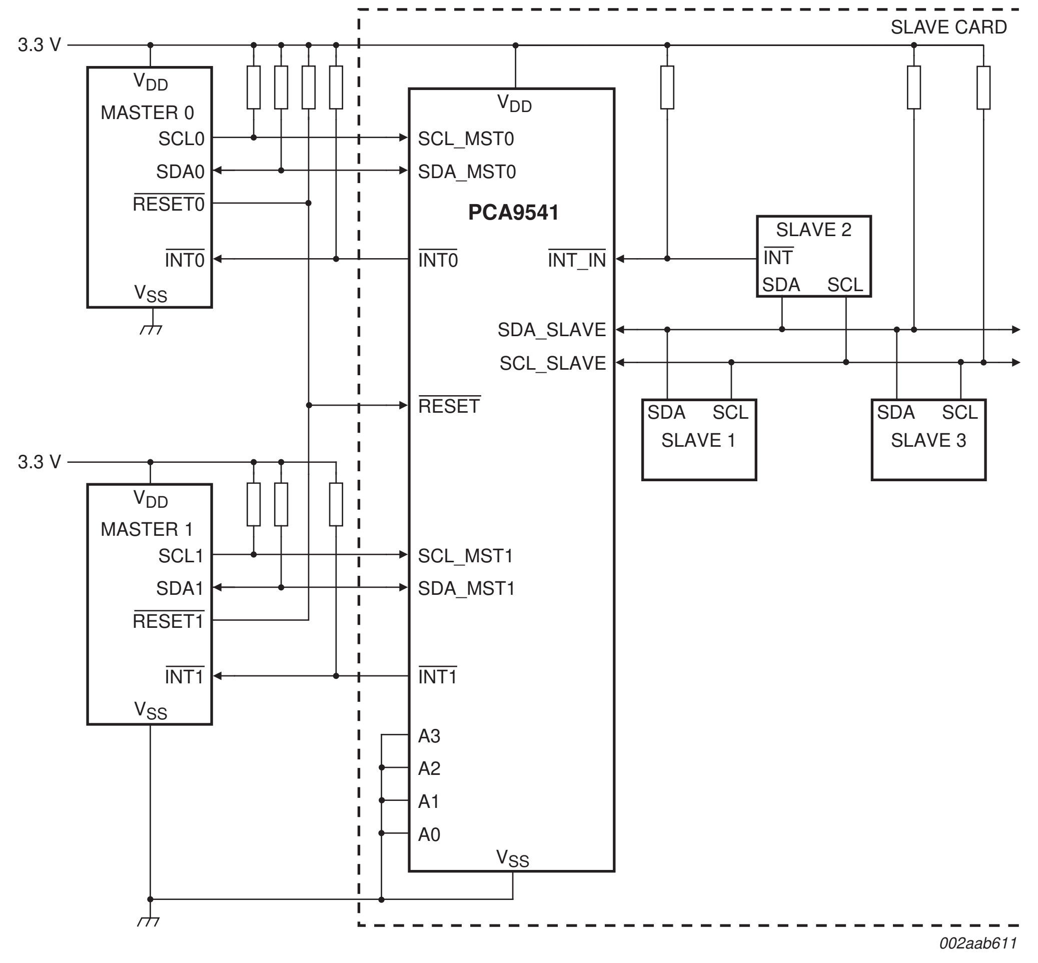


Fig 17. Typical application

10.1 Specific applications

The PCA9541 is a 2-to-1 I²C-bus master selector designed for dual master, high reliability I²C-bus applications, where continuous maintenance and control monitoring is required even if one master fails or its controller card is removed for maintenance. The PCA9541 can also be used in other applications, such as where masters share the same resource but cannot share the same bus, as a gatekeeper multiplexer in long single bus applications or as a bus initialization/recovery device.

10.2 High reliability systems

In a typical multipoint application, shown in Figure 18, the two masters (for example, primary and back-up) are located on separate I²C-buses that connect to multiple downstream I²C-bus slave cards/devices via a PCA9541/01 for non-hot swap applications to provide high reliability of the I²C-bus.

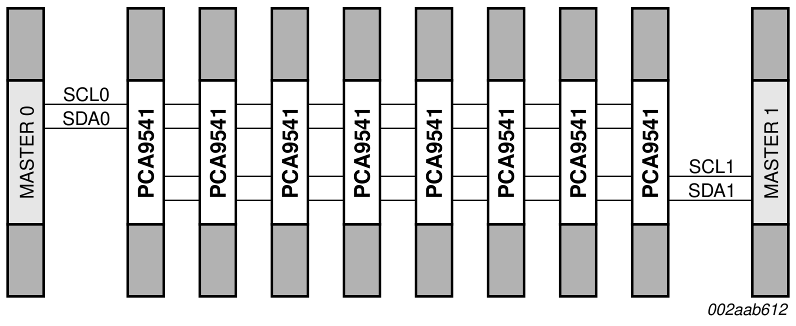


Fig 18. High reliability backplane application

I²C-bus commands are sent via the primary or back-up master and either master can take command of the I²C-bus. Either master at any time can gain control of the slave devices if the other master is disabled or removed from the system. The failed master is isolated from the system and will not affect communication between the on-line master and the slave devices located on the cards.

For even higher reliability in multipoint backplane applications, two dedicated masters can be used for every card as shown in Figure 19.

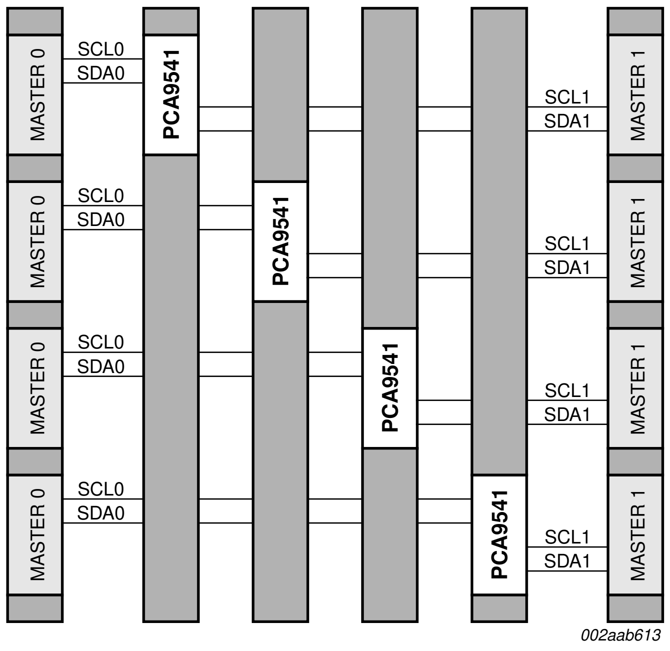


Fig 19. Very high reliability backplane application